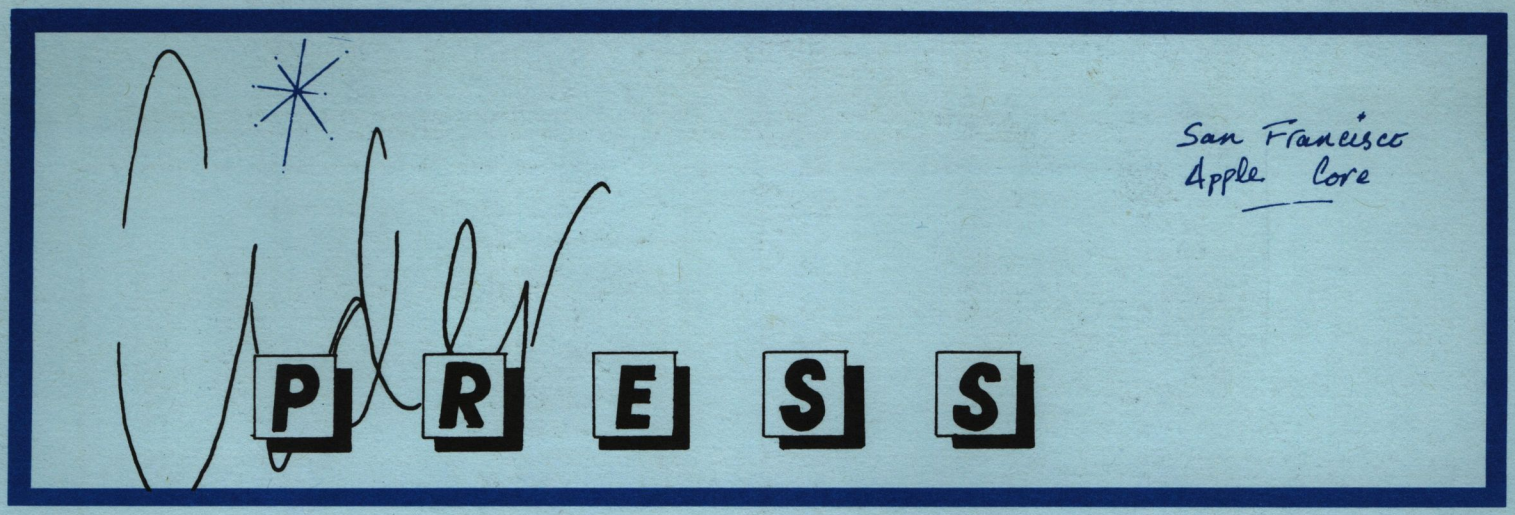


San Francisco Apple Core

IN THIS ISSUE

B.A.S.I.C. - Joyce Conklin 2
Pascal
Interrupts, Part 2 - Max J. Nareff 5
A.S.E. Macros - Max J. Nareff 6
Reviews
AppleWorks - Joyce Conklin 7
Jotto - Joyce Conklin 8
Apple Magazines - Michael Fischer 9
F.M.L.I. - Bryan Costales 13
The Call... - Kathy Rudovsky 22
Marketing - Don Scellato 23
Macinations - Don Scellato 24
VisiCalc Interest Groups - Don Scellato 25
Programming Contest - Don Scellato 25
Nu'ers - Kathy Rudovsky 26
Calendar 27
Interest Group Leaders 28

April, 1984

GENERAL MEETINGS:

May 19

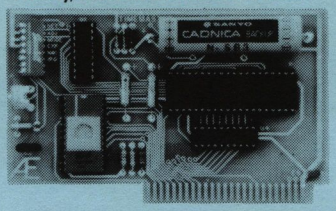
June 23

APPLIED ENGINEERING Quality Starts On The Drawing Board....

But it doesn't end there. Our commitment to excellence applies all the way down the line, through manufacturing and testing to our three year no-hassle warranty. The aim, of course, is to provide you with the best designed and best crafted hardware available. Try any of the peripherals below and see what it's like to own the finest.

The TIMEMASTER
Finally, a clock that does it ALL!

Automatically date stamps files with PRO-DOS



Fully PRO-DOS compatible

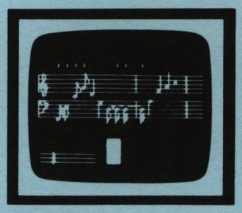
- Just plug it in and your programs can read the year, month, date, day, and time to 1 millisecond! The only clock with both year and ms.
- A rechargeable NiCad battery will keep the TIMEMASTER running for over ten years.
- Powerful 2K ROM driver — No clock could be easier to use.
- Full emulation of most other clocks, including Mountain Hardware's Appleclock (but you'll like the TIMEMASTER mode better).
- Basic, Machine Code, CP/M and Pascal software on 2 disks!
- Eight software controlled interrupts so you can execute two programs at the same time (many examples are included).
- On-board timer lets you time any interval up to 48 days long down to the nearest millisecond.

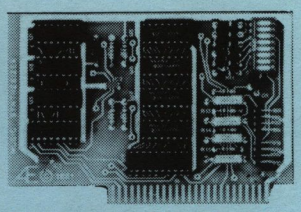
The TIMEMASTER includes 2 disks with some really fantastic time oriented programs (over 25) plus DOS dater so it will automatically add the date when disk files are created or modified. The disk is over a $200.00 value alone — we give the software others sell. All software packages for business, data base management and communications are made to read the TIMEMASTER.

If you want the most powerful and the easiest to use clock for your Apple, you want a TIMEMASTER.

PRICE $129.00

Super Music Synthesizer
Improved Hardware and Software

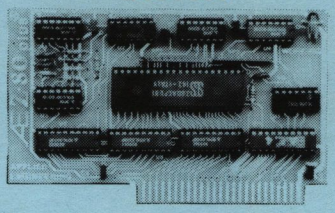




- Complete 16 voice music synthesizer on one card. Just plug it into your Apple, connect the audio cable (supplied) to your stereo, boot the disk supplied and you are ready to input and play songs.
- It's easy to program music with our compose software. You will start right away at inputting your favorite songs. The Hi-Res screen shows what you have entered in standard sheet music format.
- Now with new improved software for the easiest and the fastest music input system available anywhere.
- We give you lots of software. In addition to Compose and Play programs, 2 disks are filled with over 30 songs ready to play.
- Easy to program in Basic to generate complex sound effects. Now your games can have explosions, phaser zaps, train whistles, death cries. You name it, this card can do it.
- Four white noise generators which are great for sound effects.
- Plays music in true stereo as well as true discrete quadraphonic.
- Full control of attack, volume, decay, sustain and release.
- Will play songs written for ALF synthesizer (ALF software will not take advantage of all our card's features. Their software sounds the same in our synthesizer.)
- Our card will play notes from 30HZ to beyond human hearing.
- Automatic shutoff on power-up or if reset is pushed.
- Many many more features.

PRICE $159.00

Z-80 PLUS!



- TOTALLY compatible with ALL CP/M software.
- The only Z-80 card with a special 2K "CP/M detector" chip.
- Fully compatible with microsoft disks (no pre-boot required).
- Specifically designed for high speed operation in the Apple IIe (runs just as fast in the II+ and Franklin).
- Red "CP/M WORKING" LED indicator, the Z-80 Plus does not interfere with non-CP/M programs.
- A semi-custom I.C. and a low parts count allows the Z-80 Plus to fly thru CP/M programs at a very low power level. (We use the Z-80A at fast 4MHZ.)
- Does EVERYTHING the other Z-80 boards do, plus Z-80 interrupts.

Don't confuse the Z-80 Plus with crude copies of the microsoft card. The Z-80 Plus employs a much more sophisticated and reliable design. With the Z-80 Plus you can access the largest body of software in existence. Two computers in one and the advantages of both, all at an unbelievably low price.

PRICE $139.00

Viewmaster 80

There used to be about a dozen 80 column cards for the Apple, now there's only ONE.

- TOTALLY Videx Compatible.
- 80 characters by 24 lines, with a sharp 7x9 dot matrix.
- On-board 40/80 soft video switch with manual 40 column override
- Fully compatible with ALL Apple languages and software—there are NO exceptions.
- Low power consumption through the use of CMOS devices.
- All connections on the card are made with standard video connectors.
- All new design (using a new Microprocessor based C.R.T. controller) for a beautiful razor sharp display.
- The VIEWMASTER incorporates all the features of all other 80 column cards, plus many new improvements.

	PRICE	BUILT IN SOFTSWITCH	SHIFT KEY SUPPORT	LOW POWER DESIGN	THREE YEAR WARRANTY	7x9 DOT MATRIX	LIGHT PEN INPUTS	40 COLUMN OVERRIDE	INVERSE CHARACTERS
VIEWMASTER	**169**	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**
SUPRTERM	MORE	NO	YES	NO	NO	NO	NO	YES	YES
WIZARD80	MORE	NO	NO	NO	NO	YES	NO	YES	YES
VISION80	MORE	YES	YES	NO	NO	YES	NO	NO	NO
OMNIVISION	MORE	NO	YES	NO	NO	NO	NO	YES	YES
VIEWMAX80	MORE	YES	YES	NO	NO	YES	NO	NO	YES
SMARTERM	MORE	YES	YES	NO	NO	NO	YES	YES	NO
VIDEOTERM	MORE	NO	NO	YES	NO	YES	YES	NO	YES

The VIEWMASTER 80 works with all 80 column applications including CP/M, Pascal, WordStar, Format II, Easywriter, Apple Writer II, VisiCalc, and all others. The VIEWMASTER 80 is THE MOST compatible 80 column card you can buy at ANY price!

PRICE $169.00

MemoryMaster IIe 128K RAM Card

- Expands your Apple IIe to 192K memory.
- Provides an 80 column text display.
- Compatible with all Apple IIe 80 column and extended 80 column card software (same physical size as Apple's 64K card).
- Can be used as a solid state disk drive to make your programs run up to 20 times FASTER (the 64K configuration will act as half a drive).
- Permits your IIe to use the new double high resolution graphics.
- Automatically expands Visicalc to 95 K storage in 80 columns! The 64K config. is all that's needed, 128K can take you even higher.
- PRO-DOS will use the MemoryMaster IIe as a high speed disk drive.
- Lowest power consuming 128K card AVAILABLE.
- Complete documentation included, we show you how to use all 128K.

If you already have Apple's 64K card, just order the MEMORYMASTER IIe with 64K and use the 64K from your old board to give you a full 128K. (The board is fully socketed so you simply plug in more chips.)

MemoryMaster IIe with 128K **$249**
Upgradeable MemoryMaster IIe with 64K **$169**
Non-Upgradeable MemoryMaster IIe with 64K **$149**

Our boards are far superior to most of the consumer electronics made today. All I.C.'s are in high quality sockets with mil-spec. components used throughout. P.C. boards are glass-epoxy with gold contacts. Made in America to be the best in the world. All products work in the APPLE IIE, II, II+ and Franklin. The MemoryMaster IIe is IIe only. Applied Engineering also manufactures a full line of data acquisition and control products for the Apple; A/D converters and digital I/O cards, etc. Please call for more information. All our products are fully tested with complete documentation and available for immediate delivery. All products are guaranteed with a no hassle **THREE YEAR WARRANTY.**

All Orders Shipped Same Day
Texas Residents Add 5% Sales Tax
Add $10.00 If Outside U.S.A.
Dealer Inquiries Welcome

Send Check or Money Order to:
APPLIED ENGINEERING
P.O. Box 470301
Dallas, TX 75247

Call (214) 492-2027
7a.m. to 11p.m. 7 days a week
MasterCard, Visa & C.O.D. Welcome
No extra charge for credit cards

CIDER PRESS

Techie's Paradise

Imagine, if you will, a 30' X 30' room filled with a tremendous assortment of computers, disk drives, printers, monitors, and related peripherals. Though most of the micros are Apples, a variety of other manufacturers are also represented.

All of the equipment is "previously owned" and needs to be tested, diagnosed, and (sometimes) repaired before it can be sent on its way to a new user.

If you'd know your way around this equipment, and could keep things organized, then there may be an opportunity for you in downtown San Francisco with United States Leasing Corporation.

Compensation is negotiable, and hours are flexible. To find out more, please contact Icek Mozes at:

By Letter:	U. S. Leasing Corp. 633 Battery St. - MS515 San Francisco, CA 94111
By E-Mail:	Compu-Serve 75755,1552
By Phone:	Office: 415/445-7598 Home: 415/239-4105

THE CORE'S CORE

DIRECTORS:

Sami Capron	415/388-8631
Joyce Conklin	415/341-6955
Michael Fischer	415/933-4435
Don Scellato	415/672-2051
Alan L Smerican	415/566-2342
Allan B. Woods	415/687-9659
Alan Morris	

MEMBER SERVICE/COMMITTEES:

Membership		
Alan L Smerican		415/566-2342
Ted Chatterton		
Cider Press -		
Editor	Allan Woods	415/687-9659
BASIC	Joyce Conklin	415/341-6955
Marketing Dir.	Don Scellato	415/672-2052
Review Group	Ben Lai	415/564-7773
General Meeting	Michael Fischer	415/933-4435
Coordinating Director	Joyce Conklin	415/341-6955
Treasurer	Alan Morris	

CIDER PRESS POLICY

All manuscripts, photographs, and other materials will be considered submitted free and released for publication. They become the property of the San Francisco Apple Core and the CIDER PRESS. Authors should clearly mark all material submitted for publication so that proper credit can be given.

The views expressed in any articles are those of the authors, and not necessarily those of the San Francisco Apple Core, Inc., the Editor, Club Officers or Directors . . . or anyone else in their right minds.

The San Francisco Apple Core Incorporated is a non-profit organization comprised of and supported by owners and users of Apple I I and Apple / / / computers. The Club is run by officers and committees, and the Club endeavors to aid Apple owners through its Program Libraries, Program Disks, meetings, and this publication.

Article Submission: The CIDER PRESS welcomes any and all articles dealing with Apple computers and their associated hardware and software. Articles may be submitted (typed or printed, double - space), or as disk text files using any of the popular word processors or text editors. Any author who has an article published in the CIDER PRESS will receive a free copy of the most current Disk - of - the - Month.

Reprints: All articles appearing in the CIDER PRESS not copyrighted by the author may be reprinted by other *non - profit* Apple User Groups, without prior permission, so long as proper credit is given to both the author and the CIDER PRESS. Permission to reprint a copyrighted article must be obtained in advance, by writing to the authors, c/o the San Francisco Apple Core.

San Francisco Apple Core
1515 Sloat Blvd.
San Francisco, CA 94132

(c) Copyright 1983 by Joyce Conklin

POINTER SORT

It's time to consider another sorting routine; we had a discussion of the bubble sort algorithm (the simplest to code but slowest of the various sort routines) in the May, 1983 issue.

Also included in that issue was the listing for a Shell Sort. The **Pointer Sort** is used when you want to keep several elements associated together, but wish to sort using different criteria.

It is this type of sort, although most probably not this particular algorithm, which is used in the newer data bases such as the one included in AppleWorks (watch for a review of this integrated-software package in the near future.)

The first thing we need to do is to load the arrays with the data; in the 1 Jan. 1984 issue of the Sunday paper there was an interesting article on the cost of living in various big cities around the world.

I thought on reading it that the information would be interesting to use for this Pointer Sort problem. While the study includes 23 cities, we will limit our discussion to five, so that we can type in the information and run the program so that we can see the results without spending all night doing so.

The data is as follows:

City	Phone Call	Postage /Letter	Apt. 1 Mo.	Noodle Soup
Tokyo	$0.04	$0.25	$1044	$2.51
New York	0.20	0.20	1100	4.00
Mex.D.F.	0.001	0.04	501	1.54
Kuwait	0.07	0.05	2389	10.24
Milan	0.07	0.27	568	0.74

All prices are given in U.S. dollars.

It might be an interesting twist to the program, to assign a value for coming in first, second, etc., and find out which city ranks highest over-all in cost of living. Interestingly enough, although Tokyo has high prices on many things, it is not the highest in any one of the categories.

[The information for this problem comes from a study done by the Mitsubishi Corp.] At any rate, the problem is this -- sort the data from highest to lowest according to phone call, postage for a letter, one month's apartment rent or a bowl of soup, depending upon the selection by the user.

This means that you will need to print out the names of the cities each time, and the cities need to be "moved" along with the selected data.

To move the cities and the data in one other selected array would take a lot of time, so we will have a Pointer array which will actually move and which will serve as an index when we print out the data.

I will outline the problem this month and print the solution next month; that will give you time to work on the problem yourself. It is strongly suggested that you prepare a flow chart first, before starting to code (see the April 1983 issue for a discussion of flow-charting.

A long and detailed flow chart accompanied the article on a simulated Craps game - Nov. 1983.) You will need to load the arrays, one each for city name, phone call, postage, apartment rent, bowl of soup, *and* the pointer/index array.

Because, for our example, none of the arrays will have more than 10 items, you will not need to dimension these arrays; if you plan to use this program for more general inforation, don't forget that Applesoft requires DIMensioning for all arrays with more than 10 elements.

The first five arrays ("city" through "soup") will most easily be filled by using READ...DATA; that way all five can be loaded with one pass through a FOR...NEXT loop. In the same loop, but without READing anything, you can load the pointer array with the numbers 1-5 by using the loop variable. E.g.

```
99 REM Initialization of the arrays
100 FOR L = 1 TO 5
110 P(L) = L
120 READ CT$(L), PH(L), LT(L), AP(L),
    NS(L)
130 NEXT L
```

On the other hand, you may wish to load the arrays one at a time, with successive FOR...NEXT loops, e.g.

```
100 FOR L = 1 TO 5: P(L) = L: NEXT
110 FOR M = 1 TO 5: READ CT$(L): NEXT
120 FOR N = 1 TO 5: READ PH(L): NEXT
 etc.
```

Notice that it is not necessary to specify the loop variable in the NEXT statement, but... it's a good idea if the NEXT is separated from the FOR by several program lines.

If you choose the first method, your data must be entered as city, phone, postage, apartment rent, and soup (numbers matching each city). If you choose the second method, it must be entered as all cities on one data line, all phone rates on a second line, etc.

You must be sure in this method that you don't get some of the numbers crossed up. Using the format of the data in the sample list, this means that the first method requires entering data line by line; using the second method requires entering the data column by column.

Now, how do you sort? You will first allow the user to select the sort criterion. You need some method of selecting which comparison to make since all the data is in separate arrays; probably have your sorting routine in repetetive subroutines and use the ON X GOSUB .... statement.

Another way, which will avoid quite this much extra coding, is to load the information into a two-dimensional array. If you mentally picture this array as looking like the sample data above, you will be putting all the phone ratesin the first column, the postage in the second, etc.

Don't forget that you must DIMension *all* multi-dimensional arrays. Keep the city names in a separate array, as they are strings while the others are numerical data. Convention in Minimal BASIC says that you will DIMension your two-dimensional array rows first and then columns, so for the data as shown above you would code..

```
100 DIM A (5,4)
110 FOR R = 1 TO 5
120 FOR C = 1 TO 4
130 READ A(R, C)
140 NEXT C
150 NEXT R
```

To use this method, have the data in the form of phone, letter, apartment, soup, phone2, letter2, etc. Then have the five cities, which you will load into a separate array.

```
160 FOR L2 = 1 TO 5: READ CT$(L2): NEXT
```

You could combine line 160 above with the statement required to load the pointer array as follows:

```
160 FOR L2 = 1 TO 5: READ CT$(L2):
    P(L2) = L2: NEXT
```

Now, for the actual sort. Suppose that the selection is made to sort on phone costs, which are stored in "column 1" of the array A. Your "menu" choices should read something like...

```
WHAT DO YOU WANT TO SORT?

<1> PHONE RATE
<2> LETTER COST
<3> APARTMENT RENT
<4> BOWL OF SOUP
```

Then INPUT or GET the selection into a variable (let's call it C for "column").

The sort coding should then read as follows:

```
200 SW = 0
210 FOR L3 = 1 TO 5
220 IF A(C,L3) >= A(C,L3+1) THEN 240
230 T = P(L3): P(L3) = P(L3+1): P(L3+1)
= T: SW = 1
240 NEXT L3
250 IF SW = 1 THEN SW = 0: GOTO 210
```

Where did the P come from? Well, we are comparing the information in the array to obtain descending order, but we are actually sorting the numbers in the pointer array.

After all the sorting is done, then we want to print out the results. We will do it by the following code:

```
300 FOR L4 = 1 TO 5
310 PRINT CT$(L4); SPC(3);
320 FOR C = 1 TO 4
330 PRINT A(P(L4),C);SPC(2);
   (semicolon to keep it all
   on one line)
```

```
340 NEXT C
350 PRINT (to cause a line feed)
360 NEXT L4
```

This should be enough hints for this month; more to come next month.
:::

A TECHIE QUESTION!!

The other day I was working with my Rome Beauty and decided that I had time to re-install my 10-jey number pad, which had been detached for a time due to a broken wire (only one of the 14!) I finally had the new cable attached and even had it fixed so that I could detach it if I wanted.

Well, at any rate, that's what I thought I had. I proceeded to take the case off (yes, yes, I did remove the power cord first!) and separate the keyboard connector from its A7 location on the motherboard.

I then attached the interface board of the keypad, reattached the keyboard to the connector (J2 on the interface board), and put the cover back in place, plugged it in and..... No Joy! So back to the screw driver, take the case apart again, remove the keypad connections and reinstall the keyboard connector at A7, reattach the case and plug in the computer.

Works fine in 40-column Applesoft, even works with PR#3 for my Videx board to get 80-columns, BUT.... when I put in the Videx pre-boot for AppleWriter II, I could only see the first part of the pre-boot (it says "Program Loading" in 40-column mode) - the the screen went blank, and the program hung up completely.

I couldnt't even insert the AppleWriter II program disk and press the correct keys -- no response at all. Of course by this time I was a great example for the BEFORE part of an Arrid commercial, to say nothing of the azure color the air was becomming due to my verbal expressions of frustration!!

So far, this sounds like the typical problems that you encounter with the computer of our dreams. Pressing all the RAM chips (yes, with the power disconnected again) didn't do anything, nor was a call to my friendly Videx guru (thanks, Michael!) forthcoming with a solution.

At this time I gave up and went out to dinner, leaving the computer turned off and completely untouched by human hands for about 4 hours. (I don't know what my cat did in the interim, but I don't think she did any fixing of the Apple!)

When I arrived home, I gave it one more chance before I succumbed to using a word processor in 40-column mode, and..... it worked, just as if it had never been touched!! DOES ANYONE HAVE AN EXPLANATION FOR THIS GLITCH?

40- vs. 80-column preview

While we are discussing the use of the word processor, for all of you who have 80-column display capability, check into your manual and see if you can change the destination of the material you are printing from slot 1 to some other value.

If you can, changing it to slot 0 will display your material on the screen and you can then see how many pages it will take, if your columns are lined up, etc., etc. Try it, you'll (as they say) like it.

Index of BASIC Articles
from the Cider Press

Feb-Mar. 82
Listing of Computer Magazines

Apr-May. 82
REM statements and internal documentation

Jun 82
"Inside BASIC Games"(review); Extended print statements without awkward word split

Jul 82
Input error checks; string manipulation

Aug 82
Answers to beginner's questions (out-of-data error, use of disk drive, CTRL-D, keyboard macro for HOME [too bad, //e owners, you don't have one!]), timing of an empty FOR...NEXT loop

Sep-Oct 82
Screen windows (POKE 33, etc.)

Nov-Dec 82
Book reviews (Apple Backpack, Elementary BASIC, The VisiCalc Book); TAB, VTAB, HTAB

Jan 83
PEEK and POKE

Feb 83
CALLS; arrays

Mar 83
DIM statements; first comments on the poke program

Apr 83
READ...DATA, GOTO, IF...THEN; flowcharts

May 83
Bubble sorting

Jun 83
Re-RND; Pascal Triangle program; "Graded Computer Programs" (review); problems from that book

Jul 83
Perfect numbers; answer to last month's problem; graphics plotting; programming "tricks"

Aug 83
CTRL-S; solution to July's problem; grade-book problem

Sep 83
Solution to grade-book problem; discussion of PLE and the ESC key

Oct 83
Exec files/capture program; some useful subroutines; EXEC command

Nov 83
Graphics program (Apple Rose); Craps program with two-page flowchart

Dec-Jan 83-84
Computer aptitude test; calculating the value of pi (3.14...)

Feb 84
Analysis of arrays; poker game

::::::::::::::::::::::::::::::::::

PASCAL

-Max J. Nareff

INTERRUPTS - Part 2

In part 1 (Cider Press-Feb/Mar 1984), the program provided an interruption DURING an ONGOING screen display. After the HELP screen appeared, the program reverted to its former state.

Addition of PROCEDURE EXAMPLE2 to the original demonstration will now provide for a HELP screen to become available DURING the INPUT of a string.

The user of this demo may want to prepare another set of instructions, "INFOSHEET2", as outlined in part 1. Simpler yet, is to T(ransfer INFOSHEET1 to the above, thus providing two HELP sets.

Both sets of instructions should be on Drive #5, or the constant in PROCEDURE HELP changed appropriately.

Recompile and note how smoothly and rapidly HELP appears. In EXAMPLE2, a <SPACE> and a <CR> or two <CR>s will be needed for the interrupted string to reappear.

Entering "STOP" will disengage this cycle of the demo. It's a fine program with many educational features for the avid beginner. Thanks to Al Evans.

```
PROCEDURE EXAMPLE2;
VAR LINE : STRING;
BEGIN
   PAGE (OUTPUT);
   WRITELN ('This procedure
     demonstrates HELP during
     the input of a string');
   WRITELN ('from the keyboard.
     The HELP_KEY (?) can be
     pressed any time ');
   WRITELN ('during the string
     entry. Enter "Stop" when
     finished.');
   REPEAT
      WRITE ('Enter string: ');
      GET_STRING (LINE, EX2);
      WRITELN ('You entered: ',
        LINE)
   UNTIL (LINE = 'STOP') OR
      (LINE = 'stop')
END; (* EXAMPLE2 *)
```

ASE Macros

Using the Advanced System Editor Function Keys

Volition's Advanced System Editor (ASE) is a descendent of the popular screen editor native to UCSD Pascal. In addition to handling files of unlimited size, ASE exhibits many other improvements. The most interesting new feature is the provision for user-defined function keys.

When a function key is loaded with a sequence of instructions to be executed by a single command, it is said to hold a macro-instruction. Making macros is detailed below.

There are 8 function keys, summoned by ESC <1..8>. Key 8 contains the current date. Type ESC 8 while I(nserting or X(changing and the date appears.

How are macro-instructions, which may contain up to 78 characters, loaded into the other function keys?

Suppose Mr. Papp's name will repeatedly appear in a document. Just <record> it into a key by preceeding it with ESC R, ESC 1 and following it with either ESC R or ESC 1.

With a mere three extra commands, ASE has been taught Mr. Papp's name. Typing ESC 1 will now present Mr. Papp. Remember, you have to be in the I(nsert or X(change modes to produce it.

Another way to set a function key is to get into the general screen environment from SE (S(et E(nvironment).Select the"E(nter Function Key", which will reveal the contents of all eight function keys.

Press the number of the function key to be entered or altered, and using delimiters type "/Mr. Jones/". Having selected funtion key 2 to bear this instruction, typing ESC 2 will produce Mr.Jones, as many times as desired. Non-printing characters entered using this method appear as ordinal values when viewed from SE.

In order to add longevity to function keys, they can easily be converted to text forms and carried over to other editing sessions of the same file. This is done by typing "C(opy" and ESC n. Thus, "C" folowed by ESC 1 will produce "Mr. Papp!.".

At the next editing session, place the cursor on the first letter of the text form, press ESC T and ESC n and Mr. Papp has once again been enshrined in the selected function key, for repetitive appearances.

Still another method of generating a function key, is to call up what has been placed in the copy buffer by a previous I(nsert or D(elete. Suppose "Ms. Harlow" is in the copy buffer. Use the <takeup> command, ESC T, C(opy from buffer, ESC n, and Ms. Harlow will now be taken up into function key n.

Thus far, four methods of loading commands, etc. into the function keys have been described.

1: <record> ESC R, ESC n, (type contents), ESC R or ESC n.

2: <E(nter Function Key> from SE (Set Environment).

3: <takeup> from C(opy B(uffer - ESC T, CB, ESC n.

4: <takeup> from a TEXT form - Place cursor at start of TEXT form, ESC T, ESC n.

There are two ways to produce a TEXT form.

1:Use the C(opy, function key ESC n option for an already extant key.

2:Type a TEXT form with its special bar delimiter:"Mr. Jones!.". Place cursor on first letter and <takeup> ESC T, ESC n. See #4 above.

Note - When using TEXT forms in a program, they must be protected from the compiler by enclosing them within comments.

How can one tell what the contents of the current function keys are?

1:Try using each key in either the I(nsert or X(change mode.

2:Type SEE to get to Set Environment and E(nter Function Keys.

3:See #1 above. Writes a TEXT form of the fn key. The delimiter "!." will mark the end of the function key's contents.

4:There is still another method of revealing which function keys are taken-up and what they contain. This, for another session.

Note: Most alphabetic commands are case insensitive.

Adapted and abridged for *"The Cider Press"* by Max Nareff and the author R. Karpinski, Volition Systems Inc. from the latters article, "ASE MACROS" ;USUS NEWS & REPORT, #8, Sept. 83.

AppleWorks -- A Review
Apple Computer Inc.
$250.00

Copyright (c) 1984 by Joyce Conklin

AppleWorks is an integrated software package (what isn't, these days??) containing a word processor, a data base program, and a spreadsheet program. The package comes with a start-up disk (or boot disk) and a program disk, along with a disk of sample files, and two tutorial/demo disks.

The two-part program/start up is required because AppleWorks operates under ProDOS which must be installed in your computer first.

AppleWorks requires a //e computer with an 80 column card; it works best if the 80-column card also contains the additional 64K of RAM, as you would quickly run out of memory on the spreadsheet otherwise.

The major advantages to using an integrated software package are that commands used are the same for each application and you can move parts of one program into another.

The first advantage is a prime consideration when the user is new to microcomputers and may be more than a little aprehensive. Even those who use micros constantly will find a little change-over time required if they've been using VisiCalc all day and then shift to the word processor (ever strike /S to save an AppleWriter file?!)

The second advantage is primarily from the data base or spreadsheet to the word processor, rather than the other way around. In fact, you must save either the data base or the spreadsheet as an ASCII file before it can be transferred to and inserted into the word processor file.

The use of terms may take some getting-used-to for old-time computerniks, as the words used in AppleWorks are designed for the ordinary person-on-the-street.

As an example, the "saving" to an ASCII file referred to above uses the PRINT command (not SAVE) as you are printing to disk in a different file format from the one currently in use. SAVE is just within a file type (data base OR spreadsheet OR word processor).

The various commands use the Open-Apple key on the //e in combination with one other key, wherever possible mnemonically connected with the process to be executed. Open-Apple-C is used for COPY (within the worksheet currently displayed, to the "clipboard" or from the clipboad.) The clipboard is a buffer location in memory. Notice the use of the word COPY (an every-day word) instead of REPLICATE.

The program is completely menu-driven, and you could manage to find out all the ins and outs by just hacking your way through. In fact, for expeienced computer users, the need to step through several layers of menu to accomplish a particular function may be a bit annoying, but the advantages outweigh this minor inconvenience.

It is, however, advisable to work through the two demo disks first to see all of the possibilities. The demo programs are designed to allow ONLY one response - no playing around to see if you can crash the program! - but after you have seen what can be done, there is a tutorial manual to give you experience with your own typing, formulae (on the spreadsheet) and categories for the data base. You can also use the sample files on disk and just work through the actual examples in the tutorial.

When working on a long file you can save at any time by just pressing Open-Apple-S, as you need to assign a file name before you build it. Then OA-S is all that's required, not the whole name, or even an = sign.

OA-S will save over the previous version of the file. OA-N will allow you to rename the file in memory so that you can save different versions to disk. AppleWorks is a bit cumbersome if you have only one disk drive, because you must swap program disk and data disk, depending upon the operation which you wish to perform at the moment.

It seems, however, that once you have done a particular operation, it is in memory for the remainder of that session and you do not need to reinsert the program disk every time. Obviously it would be more convenient to have two disk drives, so that your program disk could be in drive 1 and your data disk in drive 2.

I found it more convenient to back-side my program disk with the boot-up ProDOS program; that way I only have one disk to worry about. You only need the boot-up once per session, but it saves having to move disks in and out of both the disk drive and the paper jackets.

In an attempt to replicate - ooops! COPY - formulas on the spreadsheet I found a limitation, in that you can only replicate for 62 columns. No matter how

hard I tried, with various strategems, I could not get the program to accept more than 62 columns based upon cell A1, even though it was long before the stage of running out of memory.

My spreadsheet memory test is a simple-minded one based upon the old chess-board puzzle of doubling the number of grains of wheat on each succeeding square. Except I don't get greedy - I just add one (+1) to each succeeding square or cell, rather than doubling.

OK - so now I have 62 in cell BJ1 and the copying stops, giving me back my original location of B1. When I move the cursor to BJ1 (using the OA-F - for FIND keyboard macro) and try to continue the copying from that point, it's no joy. Is that a bug, or just an undocumented feature?

When I modify the test program to work with 10 columns so that all are visible on the screen, I can copy and add down to about 200+ rows before memory is exhausted and the program hangs. Not that you have to turn off the machine and start over again, but you do have to ESC out to the menu and delete that file from the desktop.

You may have up to six files active on the desktop at one time (depending of course on available memory) and switch quickly - very quickly - from one to another. When you return to a file with which you were previously working, the cursor will be right where it was when you left the file.

Very convenient if you need to check some information in a data base while you are writing a letter or book with the word processor.

In setting up the data base you need to specify each field name, as is true with most data bases. Once you have decided what fields you are going to use, you may then start to add in the information.

Each field comes up as needed; you may press <Return> if you want to leave that field blank. When you press Open-Apple-F (for Find) to search for particular records, you will see a list of the first two to four fields (depending upon field length) of the first fifteen to twenty records.

Open-Apple-Z (for Zoom) will toggle between the multiple-record display and the display of all fields of one record, even those which don't appear on the screen in the multiple-record format. You may use any combination of search criteria, with Boolean operators (AND, OR, etc.).

The search procedure is very fast and will include specifiers from any field you want, not just the key-fields (as in PFS.)

To become thoroughly familiar with this program will take some time; pretty soon someone will provide a "command tree" showing the various layers or branches of the menus for each of the keyboard macros.

But the effort required will pay off handsomely in the increased ability to accomplish tasks which have previously required much disk-swapping, in addition to allowing many manipulations which were previously impossible.

JOTTO
Word Antics for Agile Minds
Word Associates
55 Sutter St., Suite 361
San Francisco, CA 94104

A Review
(c) Copyright 1984 by Joyce Conklin

During the game fest which was held last December, one of the items on display was a wonderful word game which I had seen advertised but had not yet been able to find in the stores because it was so new.

I am always on the lookout for games which can be played against the computer, and this is definitely one, as it doesn't support two players (except the computer as player #2!)

The game is similar to one I used to play to while away the time waiting for various test tubes to boil etc. when I worked in a biochemistry lab, and which was called in that incarnation "Keyword".

The object of the game is to guess the other player's word first. Jotto is designed to play with 5-letter words; a valuable upgrade would be to make it possible to play with 4-letter words as well, although it would probably require expanded memory and perhaps a second data disk.

You type in a guess of the computer's secret word; it responds with a number representing how many letters in your guess are also in the secret word. Not, mind you, in what position, as in the plastic-peg game of Mastermind; just that some of your letters are somewhere in the secret word! For multiple occurances of a letter, count one-on-one matches.

For example, if the secret word is BLESS, and the guess is SISSY, the score is 2; on the other hand, if the guess were SUGAR, the score would be 1 (SUGAR contains only one of the S's in BLESS - doesn't matter which one, since you are not scoring for position.)

If you finally "stump" the computer, be prepared to be told that you cheated, as the usual reason for this result is that you didn't score properly when the computer was trying to guess your word!

The disk comes with a Jotto Scratch Pad on which you may keep score, both of the letters you have eliminated (by scoring 0 for a guess word) and of your secret word.

This is an excellent practice game for improving your Scrabble (R.) vocabulary, as well as an enjoyable game in itself for word buffs.

Apple Magazines

[Note: This article originally appeared, in slightly different form, in three parts in *Bay Area Computer Currents*.]

By Michael Fischer

Apple users, like all personal computer users, have questions they can't answer or problems they can't solve. They often need more information than comes in the computer manuals and turn to the many general circulation computer magazines.

For some, the information in the general coverage magazines is enough with the occasional article on the Apple.

Apple owners are blessed, however, with at least 11 magazines which focus exclusively or significantly on the Apple. The real problem is chosing from among this richness.

Call -A.P.P.L.E.: This magazine is actually a users group newsletter, although many members live far from the club's home in Seattle. It covers only Apples and features a wide variety of articles. There are how-to-do-its; programming hints; programs in BASIC, Pascal and 6502 Assembler; and question and answer columns. Articles also cover such subjects as spreadsheets, CP/M, and telecommunications. Announcements of new Apple products and several long reviews usually appear in each issue. It is also one of the few publications that does comparative reviews.

Call -A.P.P.L.E. has been published for nearly six years. It usually has something in it for every Apple user. Its writers are literate and knowledgeable. The magazine also features member software and other products, and often articles feature these products or make use of them.

This is one magazine that most Apple users should consider getting. It emphasizes the active user and programmer and some material is aimed at the more experienced user. Many articles can be a struggle for the novice, but the effort spent will be rewarded with significant and useful information.

A recent issue had 100 pages, about 31% advertising.

To subscribe to *Call -A.P.P.L.E.* you must join A.P.P.L.E. Their initiation fee is $25.

Call -A.P.P.L.E. is published monthly and includes a reader service card. Single issues are $2.50; subscriptions are $20.00 a year. A.P.P.L.E., 21246 68th Ave. S., Kent, WA 98032.

Micro, subtitled "Advancing Computer Knowledge": This magazine focuses on one subject each issue. It covers Apple, Atari, Commodore Vic 20, and Commodore 64, and the TRS Color Computer.

There are several general articles on each month's topic as well as several machine specific articles for each computer. Program listings are given for each of the covered computers.

Each computer has its own section in the magazine. There are columns on new Apple items, both hardware and software, and a couple of short, highly structured reviews of Apple products.

Micro has been published for over 5 years. It began as a magazine devoted to the 6502 microprocessor (the chip in the Apple) and was the magazine for 6502 machine code hackers. Later it tried to cover all 6502 based computers and still later 6809 chip machines as well.

Micro currently is aimed at the general home user, both experienced and inexperienced. Its utility to most Apple owners is marginal due both to its multi-computer coverage and its changing editorial focus.

A recent issue had 116 pages, about 30% advertising.

Micro is published monthly. Single issues are $2.50; subscriptions are $24 a year. Micro, P.O. Box 6502, Amherst, NH 03031.

Nibble, subtitled "The Reference to Apple

Computing": Nibble is the Apple programmer's magazine. As has been true almost from its inception nearly four years ago, the magazine features articles which contain a complete working program, together with any necessary machine language enhancements. Each article usually includes a full description of the program and a detailed analysis of how the program functions.

Programs are usually written in Applesoft, although the magazine has also featured programs in 6502 Assembler and Pascal.

One of the "trademarks" of *Nibble* is the long program listing extending over a number of pages and printed in small type. The programs in each magazine are avaiable on one or two disks at a cost of about $20-$30 each.

Software writers and sellers are a major *Nibble* audience. The Editor/Publisher's column is straight from "How to succeed in business by really trying"; a column on the legal aspects of producing software appears in each issue. Other features include short how-to-do-it articles, new product announcements and regular columns on Applesoft, machine language and graphics. There is also an occasional in-depth product review.

Nibble is the magazine for the programmer, especially the Applesoft and commercial programmer.

A recent issue contained 204 pages, about 58% advertising.

Nibble is published eight times a year and includes a reader service card. Single issues are $3.25; subscriptions are $19.95 per year. MicroSPARC, Inc., 10 Lewis St., Lincoln, MA 01773.

in Cider, subtitled "Green's Apple Magazine": Now approaching its first anniversary, *in Cider* continues its attempt to please both the user and programmer. The magazine comes from the publisher of *Microcomputing Magazine*.

Each issue contains several columns on such subjects as graphics, Applesoft, business, assembly language, and the Apple ///. There is also a question and answer column, new products section, and a number of short articles under the general heading of hints and techniques.

A substantial portion of each *in Cider* contains a number of articles presenting home application programs such as educational uses, graphics or programming utilities. The programs and the explanatory articles are generally shorter than those in *Nibble*. Programs are in BASIC and Pascal. *in Cider* is rare among Apple magazines in that it often includes construction articles such as modifications to the disk drive or building your own joystick.

The other major portion of *in Cider* consists of reviews. Some 10 to 20 programs, several hardware products, and even some books are covered in medium length articles.

A recent issue had 260 pages, about 49% advertising.

in Cider is published monthly and includes a reader service card. Single issues are $2.95; subscriptions are $25 a year. Wayne Green Inc., 80 Pine St., Peterborough, NH 03458.

Apple Orchard, subtitled "The Premier Magazine for Apple Computer Users": This magazine is the publication of the International Apple Corps, a federation of Apple Users Groups. In many ways it is a users group newsletter written large.

Over its four years of publication, this magazine has evolved from a sporadically produced, unfocused publication with uneven quality to the present high quality magazine with a little of something for everyone. By reprinting the best material from user group newsletters, it is often on the cutting edge of Apple developments.

Recently *Apple Orchard* has had one or more articles on the use of Apple computers in a particular area or concerning a person making an unusual use of an Apple. Its feature articles range from an interview with a computer graphic artist to an investigation of the power of the Word Processing Language utility in AppleWriter II.

Each issue includes an editor's column, usually a philosophical view of Apple developments, and a column from the International Apple Core. Its description of new products is among the largest in all of the Apple magazines. The articles are often surprising, usually informative and of interest to a variety of Apple owners.

A recent issue had 116 pages, 45% advertising.

Apple Orchard is published monthly and includes a reader service card. Single issues are $3.25; subscriptions are $24 a year. Apple Orchard, 908 George Street, Santa Clara, CA 95050.

Softalk is one of the most interesting Apple magazines from the standpoint of development. It began as a free publication to Apple owners that was filled with advertising and columns about people in the software publishing business. Early readers of *Softalk* remember the poor quality covers (which the postal service routinely ripped) and its small size. *Softalk* is now probably the largest Apple publication in terms of number of pages.

Each issue features an article about an Apple peripheral or software company, a guide to new and classic Apple programs, a

column on industry news, mini-reviews of new programs, and a question and answer column. Its letters column has one of the liveliest ongoing Apple discussions.

Softalk has had a number of series on particular Apple topics. Its first series of columns, Assembly Lines, has been collected into a very useful book on Apple assembly language. Current topics include Applesoft routines, beginners, investing, spreadsheets, DOS, assembly language, Applesoft techniques, graphics, telecommunications, business, Pascal, education and Apple CP/M. There is often an article on hardware and construction projects.

Readers are challenged each month with a *Softalk* contest. Winners are awarded merchandise certificates.

Each issue includes a listing of the best-selling Apple programs in a variety of categories -- Apple ///, arcade games, word processors, home education, adventure games, strategy games, fantasy games, business, hobby (utilities), and home -- as well as the top 30 bestselling programs.

A recent issue had 420 pages, 48% advertising.

Softalk is published monthly. Single issues are $3.00; subscriptions are free for one year, then $24 a year. Softalk, P.O. Box 60, North Hollywood, CA 91603.

Peelings II, subtitled "The Magazine of Apple Software and Hardware Evaluation": Each issue contains about 10 program reviews that follow a similar format. The reviews evaluate the usefulness of the product. They do not follow a tutorial approach. In addition, each program is assigned an overall letter grade ranging from AAA down to F, including pluses and minuses. The letter grade is a summary of the overall success of the program and considers similar products, ease of use, documentation, and sophistication.

Each issue of *Peelings* focuses on one or two areas such as spreadsheets, education, data bases, or home entertainment. Similar products are reviewed by different people in separate reviews. This is not as useful as if all programs were compared by the same person in one article. However, *Peelings* does print comparison charts. In addition the magazine develops standards for evaluating certain types of programs such as data base managers and word processors. Reader reaction to proposed standards is encouraged. There are occasionally multiple reviews of the same product.

Responses from the product manufacturer are also published in *Peelings*, either with the review itself or in a subsequent issue. The number of responses, especially from companies given fairly good reviews, attests to the integrity and quality ascribed by the companies to the magazine's reviews.

In addition to reviews and discussions on how to evaluate programs, *Peelings* contains a few other articles on the Apple as well as a letters column.

A recent issue had 72 pages, 28% advertising.

Peelings is published nine times a year. Single issues are $3.00; subscriptions are normally $21 a year although a holiday special currently has subscriptions for $18 a year. Peelings, P.O. Box 188, Las Cruces, NM 88001.

Softside, subtitled "The Magazine for You and Your Computer: This magazine suffers either from not knowing what it wants to be or from trying to be too many different things. It currently features Atari, IBM PC and TRS-80 as well as the Apple.

Each issue begins with some general features. These range from think-pieces on computers and society to use of financial spreadsheets to adventure games. The quality of the articles is as varied as the subject matter.

Each of the featured computers has its own section of the magazine. That section contains two or three articles on the specific computer, usually a product review and a programming discussion. Inserted in each magazine is a pamphlet called "Selections," which contains the listing for the featured program plus an additional program or two, all written for that particular computer. If you buy *Softside* at the newsstand, however, "Selections" only contains a listing of the featured program for each covered computer.

Rounding out each issue is a letters column, an enumeration of "bugs" discovered in previously published programs, and new product announcements.

A disk version of *Softside* is also available. It consists of the magazine plus a disk with the programs featured that month as well as an extra "bonus" program.

A recent issue had 80 pages, 35% advertising, plus the program listing insert.

Softside is published monthly. Single issues are $3; subscriptions are $18 for 6 months, $30 for one year. The disk version is $16 a month; $79 for six months, $149 for one year. Softside, 6 South Street, Milford, NH 03055.

A+, subtitlted "The Independent Guide to Apple Computing": This magazine debuted in late 1983. Three issues have been printed to date. The opinions expressed about it have to be viewed with caution as the magazine is still in its formative stage.

The material is mainly for the user rather than the programmer/hobbyist. Each issue focuses on a particular subject; features so far have been on graphics and finances. It has an introductory article and several other pieces on that subject, usually including comparative reviews of products and articles discussing how Apples are being used in applications related to that topic.

A+ has ongoing columns on graphics, dBase II, and reader questions. The magazine also reviews products not related to the monthly topic. Other features include product announcements, a game reviews, and brief news items.

The magazine is published by Ziff-Davis Publishing Company, which also prints *Creative Computing*. Many of the reviews of individual products feature a "report card" summary developed by Editor Maggie Cannot when she was at *Infoworld*. It would be helpful if all reviews contained this summary. Reviews also don't make it clear whether the product is for the][+, *//e* or *///*.

The magazine has seen steady improvement since its somewhat disappointing first issue.

Some Apple magazines say they cover the Apple *///* and Lisa. *A+* is one of the few magazines that lives up to that claim.

A disk version of the magazine is planned for March.

A recent issue had 204 pages, 48% advertising.

A+ is published monthly and includes a reader service card. Singe issues are $2.95; subscriptions are $24.97 a year with a current special of $19.97 a year. A+, P.O. Box 2964, Boulder, CO 80321.

Softline: This magazine, a spinoff from *Softalk*, was originally about Apple games. It now claims to cover games on a wide variety of machines although it tends, not surprisingly, to concentrate on the Apple in addition to the Atari, Commodore 64, Vic 20, and to a lesser extent the TRS-80 and IBM PC.

As befits a magazine about games, there are usually several contests in each issue. There are some fairly short game reviews, which clearly indcate which game runs on which computer. The magazine also has a column featuring reader questions, clues and answers that focus on how to solve certain problems in the various fantasy and adventure games.

Softline has several articles each issue on game software companies and the game software business. On-going series currently include how to write a game in 6502 assembler (for both the Apple and the Commodore 64) and adventure game programming as well as producing sounds on the Atari. Prior series topics have included information about Apple high resolution graphics.

Articles about computer games often suffer from terminal cuteness. Unfortunately (or fortunately, depending on your taste), *Softline* is not immune from this style of writing.

No dicussion of *Softline* would be complete without mention of its high scores column, a compendium of the highest score claimed in different games together with a lively discussion of who did what and how. There is on-going debate as to whether some claimants "cheat". Selected high scorers are given prizes. There are a variety of rules for submitting and verifying high scores and permanently retiring the high score for a particular game.

A recent issue had 72 pages, 32% advertising.

Softline is published bimonthly. Single issues are $2; subscriptions are $12 per year. They used to and may still have free subscriptions for new computer owners. Softline, 7230 Laurel Canyon Blvd., Box 60, North Hollywood, CA 91603 (213) 980-5074.

Apple Assembly Line: If you don't program in 6502 assembly language and have no desire to, then this is definitely not your magazine. This publication is not for the casual user or even for the beginning assembly language programmer; it assumes a fair amount of knowledge. However, if you are more than a beginner in asembly language programming, this publication should be on your regular reading list. It provides a great deal of valuable information not found anywhere else.

The magazine is published by S-C Software Corporation; S-C is from the last name of Bob Sander-Cederlof, one of the major writers about Apple assembly language programming. His company also sells a very high quality and versatile assembler, which has macro capabilities and other advanced features including conditional assembly.

Apple Assembly Lines was originally started as a support newsletter for the S-C Assembler. Now in its fourth year, it is the only major periodical devoted to Apple 6502 assembly language programming.

Even though the magazine is not exclusively for users of the S-C assembler, the programs make extensive use of the special features of that assembler. Therefore the reader either needs that assembler or needs the expertise to "work around" those features in another assembler. Most of the articles give advcie for those who don't have the S-C Assembler.

An index to the first three years of arti-cles was printed in a recent issue. Re-cent issues have begun to print the first disassembled listings of ProDOS, Apple's soon-to-be released new disk operating system.

The advertisements and products mentioned in this publication differ from those found in most other Apple publications. As expected, they tend to appeal to the assembly language programmer. The pages are one-half regular magazine size.

A recent issue had 28 pages, about 20% advertising.

Apple Assembly Lines is published monthly. Single issues are $1.50; subscriptions are $18 per year. S-C Software Corporation, 2331 Gus Thomassen, Suite 125, Dallas, TX 75228 (214) 324-2050.

Filemanager/Machine Language Interface

-Bryan Costales

Part 3 of a Series

(c) B. Costales 1983

ON WITH THE SHOW

This month's source listing is the final one of this series for FMLI proper. As it is so very long, I will refrain from much of a preamble. Next month I will conclude with a header file demonstrating FMLI's capabilities. It will be a small Unix(tm) style shell, with a history mechanism patterned after the Berkeley Cshell. I will also provide listings for the unfortunate errata of my lead article.

FINAL SOURCE LISTING FOR FMLI

```
0DD5                 8  ;
0DD5                 9  ;-------------------------------------------------------
0DD5                10  FMCALLX:
0DD5                11  ; Common file manager calling sequence where errors
0DD5                12  ; can legally abort to MAPFMERR, and where no further
0DD5                13  ; processing is required after the call.
0DD5                14  ;-------------------------------------------------------
0DD5 20 FD AA       15          JSR CALLFM
0DD8 90 03          16          BCC COMCLCEX
0DDA 4C B1 0B       17          JMP MAPFMERR
0DDD                18  COMCLCEX:
0DDD 20 DF 0A       19          JSR BFCLC
0DE0 4C CB 0A       20          JMP BFEXIT
0DE3                21  ;
0DE3                25  ;-------------------------------------------------------
0DE3                26  BFERROR:
0DE3                27  ; Calling format:   JSR CALLFMLI
0DE3                28  ;                   ADR ERRORNUMBER
0DE3                29  ; Where:
0DE3                30  ;         ERRORNUMBER is either the value returned by
0DE3                31  ;          a previous call to another FMLI routine, or
0DE3                32  ;          one specified by the calling program.
0DE3                33  ; Output is always to stderr. Unknown error numbers
0DE3                34  ;  output as "bad system call".
0DE3                35  ;-------------------------------------------------------
```

```
0DE3 20 09 0F    36            JSR PRFNAME          ; print "filename" v,s,d
0DE6 20 F8 0A    37            JSR PULLADR          ; fetch error number
0DE9 C9 10       38            CMP #$10
0DEB 90 02       39            BLT >1
0DED A9 0E       40            LDA #$0E
0DEF 0A          41  ^1        ASL
0DF0 AA          42            TAX
0DF1 BD E9 0E    43            LDA EMSGTBL,X
0DF4 85 FA       44            STA AP
0DF6 BD EA 0E    45            LDA EMSGTBL+$01,X
0DF9 85 FB       46            STA AP+$01
0DFB A0 00       47            LDY #$00
0DFD B1 FA       48  ^2        LDA (AP),Y
0DFF F0 06       49            BEQ >3
0E01 20 8E 0F    50            JSR ERROUT
0E04 C8          51            INY
0E05 D0 F6       52            BNE <2
0E07 A9 8D       53  ^3        LDA #CR
0E09 20 8E 0F    54            JSR ERROUT
0E0C 4C DD 0D    55            JMP COMCLCEX
0E0F             56  ;
0E0F E5 EE E4    57  EM00      BYT "end of data",0
0E1B E3 E1 EE    58  EM01      BYT "can't open",0
0E26 E3 E1 EE    59  EM02      BYT "can't close",0
0E32 E6 E9 EC    60  EM03      BYT "file protected",0
0E41 E4 E9 F3    61  EM04      BYT "disk full",0
0E4B E3 E1 F4    62  EM05      BYT "catalog full",0
0E58 F7 F2 EF    63  EM06      BYT "wrong file type",0
0E68 E2 E1 E4    64  EM07      BYT "bad file name",0
0E76 E2 E1 E4    65  EM08      BYT "bad volume number",0
0E88 C9 AF CF    66  EM09      BYT "I/O error",0
0E92 F3 F9 F3    67  EM0A      BYT "system failure",0
0EA1 E3 E1 EE    68  EM0B      BYT "can't allocate buffer",0
0EB7 F7 F2 E9    69  EM0C      BYT "write only",0
0EC2 F2 E5 E1    70  EM0D      BYT "read only",0
0ECC E2 E1 E4    71  EM0E      BYT "bad system call",0
0EDC EF F5 F4    72  EM0F      BYT "out of range",0
0EE9             73  ;
0EE9 0F 0E 1B    74  EMSGTBL   ADR EM00,EM01,EM02
0EEF 32 0E 41    75            ADR EM03,EM04,EM05
0EF5 58 0E 68    76            ADR EM06,EM07,EM08
0EFB 88 0E 92    77            ADR EM09,EM0A,EM0B
0F01 B7 0E C2    78            ADR EM0C,EM0D,EM0E
0F07 DC 0E       79            ADR EM0F
0F09             80  ;
0F09             81  ;
0F09             82  PRFNAME:
0F09 A0 1D       83            LDY #!29
0F0B B1 04       84  ^0        LDA (FCURNP),Y
0F0D C9 A0       85            CMP #" "
0F0F D0 05       86            BNE >1
0F11 88          87            DEY
0F12 10 F7       88            BPL <0
0F14 30 18       89            BMI >3
0F16 20 8C 0F    90  ^1        JSR QOUT
0F19 98          91            TYA
0F1A AA          92            TAX
0F1B A0 00       93            LDY #$00
0F1D B1 04       94  ^2        LDA (FCURNP),Y
0F1F 20 8E 0F    95            JSR ERROUT
0F22 C8          96            INY
0F23 CA          97            DEX
0F24 10 F7       98            BPL <2
0F26 20 8C 0F    99            JSR QOUT
0F29 A9 A0      100            LDA #" "
0F2B 20 8E 0F   101            JSR ERROUT
0F2E A9 F6      102  ^3        LDA #"v"
0F30 20 8E 0F   103            JSR ERROUT
0F33 A5 00      104            LDA FCURVOL
0F35 20 5C 0F   105            JSR HEXTODEC
0F38 A9 AC      106            LDA #","
0F3A 20 8E 0F   107            JSR ERROUT
```

```
OF3D A9 F3       108            LDA #"s"
OF3F 20 8E OF    109            JSR ERROUT
OF42 A5 02       110            LDA FCURSLOT
OF44 20 5C OF    111            JSR HEXTODEC
OF47 A9 AC       112            LDA #","
OF49 20 8E OF    113            JSR ERROUT
OF4C A9 E4       114            LDA #"d"
OF4E 20 8E OF    115            JSR ERROUT
OF51 A5 01       116            LDA FCURDRIV
OF53 20 5C OF    117            JSR HEXTODEC
OF56 A9 A0       118            LDA #" "
OF58 20 8E OF    119            JSR ERROUT
OF5B 60          120            RTS
OF5C             121  ;
OF5C             122  HEXTODEC:
OF5C 85 FC       123            STA VP
OF5E A9 00       124            LDA #$00
OF60 85 FD       125            STA VP+$01
OF62 A2 02       126            LDX #$02
OF64 A0 00       127  ^1        LDY #$00
OF66 A5 FC       128  ^2        LDA VP
OF68 38          129            SEC
OF69 FD 91 OF    130            SBC TENTBL,X
OF6C 90 05       131            BCC >3
OF6E 85 FC       132            STA VP
OF70 C8          133            INY
OF71 D0 F3       134            BNE <2
OF73 C0 00       135  ^3        CPY #$00
OF75 D0 08       136            BNE >4
OF77 A5 FD       137            LDA VP+$01
OF79 D0 04       138            BNE >4
OF7B E0 00       139            CPX #$00
OF7D D0 09       140            BNE >5
OF7F E6 FD       141  ^4        INC VP+$01
OF81 98          142            TYA
OF82 18          143            CLC
OF83 69 B0       144            ADC #"0"
OF85 20 8E OF    145            JSR ERROUT
OF88 CA          146  ^5        DEX
OF89 10 D9       147            BPL <1
OF8B 60          148            RTS
OF8C             149  ;
OF8C A9 A2       150  QOUT      LDA #'"'+$80
OF8E 6C F1 00    151  ERROUT    JMP (CONSOUT)
OF91 01 0A 64    152  TENTBL    .DA #!01,#!10,#!100
OF94             153  ;
OF94             157  ;-----------------------------------------------------
OF94             158  BFPUTC:
OF94             159  ; Calling format: JSR CALLFMLI
OF94             160  ;                  BYT FPUTC
OF94             161  ;                  ADR FILEDESCRIPTOR
OF94             162  ;                  ADR CHARACTER
OF94             163  ; Fetches next byte from opened file. Returns char
OF94             164  ;  in a-reg. Carry set on error. A-reg = 00 and carry
OF94             165  ;  set if EOF.
OF94             166  ;-----------------------------------------------------
OF94 20 6D OD    167            JSR GETFD1
OF97 A5 07       168            LDA FRWSTAT          ;check read/write status
OF99 D0 05       169            BNE >1
OF9B A9 0D       170            LDA #EREAD
OF9D 4C A3 0B    171            JMP AERR1
OFA0 A9 04       172  ^1        LDA #$04             ;opcode=4 write
OFA2 8D BB B5    173            STA FMOPCODE
OFA5 A9 01       174            LDA #$01             ;subcode=1 write 1 byte
OFA7 8D BC B5    175            STA FMSUBCOD
OFAA 20 F8 0A    176            JSR PULLADR
OFAD 8D C3 B5    177            STA FMBYTE           ;fetch the byte
OFB0 4C D5 0D    178            JMP FMCALLX          ;call the filemanager.
```

```
OFB3                    179  ;
OFB3                    183  ;---------------------------------------------------
OFB3                    184  BFTYPE:
OFB3                    185  ; Calling format:  JSR CALLFMLI
OFB3                    186  ;                  BYT FTYPE
OFB3                    187  ;                  ADR FILEDESCRIPTOR
OFB3                    188  ; Returns file's type in a-reg.
OFB3                    189  ;---------------------------------------------------
OFB3 20 75 OD           190           JSR GETFDO
OFB6 A5 03              191           LDA FCURTYPE
OFB8 85 F5              192           STA ASAVE
OFBA 4C DD OD           193           JMP COMCLCEX
OFBD                    194  ;
OFBD                    195  ;---------------------------------------------------
OFBD                    196  BFDATA:
OFBD                    197  ; Calling format:  JSR CALLFMLI
OFBD                    198  ;                  BYT FDATA
OFBD                    199  ;                  ADR FILEDESCRIPTOR
OFBD                    200  ; Returns address of file's data buffer in a,x.
OFBD                    201  ;---------------------------------------------------
OFBD 20 75 OD           202           JSR GETFDO
OFC0 A5 0C              203           LDA FCURDAT
OFC2 85 F5              204           STA ASAVE
OFC4 A5 0D              205           LDA FCURDAT+$01
OFC6 85 F6              206           STA XSAVE
OFC8 4C DD OD           207           JMP COMCLCEX
OFCB                    208  ;
OFCB                    212  ;---------------------------------------------------
OFCB                    213  BFTS:
OFCB                    214  ; Calling format:  JSR CALLFMLI
OFCB                    215  ;                  BYT FTS
OFCB                    216  ;                  ADR FILEDESCRIPTOR
OFCB                    217  ; Returns address of file's t/s list in a,x.
OFCB                    218  ;---------------------------------------------------
OFCB 20 75 OD           219           JSR GETFDO
OFCE A5 0A              220           LDA FCURTS
OFD0 85 F5              221           STA ASAVE
OFD2 A5 0B              222           LDA FCURTS+$01
OFD4 85 F6              223           STA XSAVE
OFD6 4C DD OD           224           JMP COMCLCEX
OFD9                    225  ;
OFD9                    226  ;---------------------------------------------------
OFD9                    227  BFGETSDV:
OFD9                    228  ; Calling format:  JSR CALLFMLI
OFD9                    229  ;                  BYT FGETSDV
OFD9                    230  ;                  ADR FILEDESCRIPTOR
OFD9                    231  ; Returns file's slot, drive, vol. in a,x,y.
OFD9                    232  ;---------------------------------------------------
OFD9 20 75 OD           233           JSR GETFDO
OFDC A5 02              234           LDA FCURSLOT
OFDE 85 F5              235           STA ASAVE
OFE0 A5 01              236           LDA FCURDRIV
OFE2 85 F6              237           STA XSAVE
OFE4 A5 00              238           LDA FCURVOL
OFE6 85 F7              239           STA YSAVE
OFE8 4C DD OD           240           JMP COMCLCEX
OFEB                    241  ;
OFEB                    245  ;---------------------------------------------------
OFEB                    246  BFSETSDV:
OFEB                    247  ; Calling format:  JSR CALLFMLI
OFEB                    248  ;                  BYT FSETSDV
OFEB                    249  ;                  ADR 3DATABYTES
OFEB                    250  ; Where 3DATABYTES are 3 consecutive bytes: slot,
OFEB                    251  ;  drive, volume.
OFEB                    252  ; Use to precondition for calls to catalog, etc.
OFEB                    253  ; NOTE: A bug in the code defaults to the last
OFEB                    254  ;  drive accessed if this is the first buffer
OFEB                    255  ;  allocated.
OFEB                    256  ;---------------------------------------------------
OFEB 20 F8 0A           257           JSR PULLADR
OFEE 85 02              258           STA FCURSLOT
OFF0 86 01              259           STX FCURDRIV
```

```
0FF2 A0 02       260            LDY #$02
0FF4 B1 FA       261            LDA (AP),Y
0FF6 85 00       262            STA FCURVOL
0FF8 4C DD 0D    263            JMP COMCLCEX
0FFB             264  ;
0FFB             265  ;-------------------------------------------------------
0FFB             266  BFSETUP:
0FFB             267  ; Calling format:  JSR CALLFMLI
0FFB             268  ;                  BYT FSETUP
0FFB             269  ; Returns the current address of HIMEM in a,x regs.
0FFB             270  ;-------------------------------------------------------
0FFB A5 FE       271            LDA HIMEM
0FFD 85 F5       272            STA ASAVE
0FFF A5 FF       273            LDA HIMEM+$01
1001 85 F6       274            STA XSAVE
1003 4C DD 0D    275            JMP COMCLCEX
1006             276  ;
1006             280  ;-------------------------------------------------------
1006             281  BFGETPOS:
1006             282  ; Calling format: JSR CALLFMLI
1006             283  ;                 BYT FGETPOS
1006             284  ;                 ADR FILEDESCRIPTOR
1006             285  ; Returns the current offset into the file in bytes,
1006             286  ;  from a base of 0. ie, position #1 =00 position #2
1006             287  ;  = 01 etc. Position returned in a,x regs. lo,hi.
1006             288  ;-------------------------------------------------------
1006 20 75 0D    289            JSR GETFDO
1009 A0 1B       290            LDY #$1B
100B B1 08       291            LDA (FCURWK),Y
100D 85 F5       292            STA ASAVE
100F C8          293            INY
1010 B1 08       294            LDA (FCURWK),Y
1012 85 F6       295            STA XSAVE
1014 4C DD 0D    296            JMP COMCLCEX
1017             297  ;
1017             301  ;-------------------------------------------------------
1017             302  BFREAD:
1017             303  ; Calling format:  JSR CALLFMLI
1017             304  ;                  BYT FREAD
1017             305  ;                  ADR FILEDESCRIPTOR
1017             306  ;                  ADR BUFFER
1017             307  ;                  ADR NUMBERtoREAD
1017             308  ; Where BUFFER is where the data will be read into.
1017             309  ;  and NUMBERtoREAD is the number of bytes, lo,hi, to
1017             310  ;  be read into that buffer.
1017             311  ;-------------------------------------------------------
1017 20 38 10    312            JSR COMRW
101A A9 03       313            LDA #$03
101C 8D BB B5    314            STA FMOPCODE
101F 4C D5 0D    315            JMP FMCALLX
1022             316  ;
1022             317  ;-------------------------------------------------------
1022             318  BFWRITE:
1022             319  ; Calling format:  JSR CALLFMLI
1022             320  ;                  BYT FWRITE
1022             321  ;                  ADR FILEDESCRIPTOR
1022             322  ;                  ADR BUFFER
1022             323  ;                  ADR NUMBERtoWRITE
1022             324  ; Where BUFFER is where the data will be written from.
1022             325  ;  and NUMBERtoWRITE is the number of bytes, lo,hi, to
1022             326  ;  be written from that buffer.
1022             327  ;-------------------------------------------------------
1022 20 38 10    328            JSR COMRW
1025 AD C1 B5    329            LDA FMRANGEL
1028 D0 03       330            BNE >0
102A CE C2 B5    331            DEC FMRANGEL+$01
102D CE C1 B5    332  ^0        DEC FMRANGEL
1030 A9 04       333            LDA #$04
1032 8D BB B5    334            STA FMOPCODE
1035 4C D5 0D    335            JMP FMCALLX
1038             336  ;
```

```
1038                    337  ;---------------------------------------------------
1038                    338  COMRW:
1038                    339  ; common initialization routine for read and write.
1038                    340  ;---------------------------------------------------
1038 20 64 0D           341           JSR GETFD2
103B 20 F8 0A           342           JSR PULLADR
103E 8D C3 B5           343           STA FMRANGEA
1041 8E C4 B5           344           STX FMRANGEA+$01
1044 20 F8 0A           345           JSR PULLADR
1047 8D C1 B5           346           STA FMRANGEL
104A 8E C2 B5           347           STX FMRANGEL+$01
104D A9 02              348           LDA #$02
104F 8D BC B5           349           STA FMSUBCOD
1052 60                 350           RTS
1053                    351  ;
1053                    355  ;---------------------------------------------------
1053                    356  BFSEEK:
1053                    357  ;Calling format:    JSR CALLFMLI
1053                    358  ;                   BYT FSEEK
1053                    359  ;                   ADR FILEDESCRIPTOR
1053                    360  ;                   ADR FILEPOSITION
1053                    361  ;                   ADR FROMBYTE
1053                    362  ; Where FILEPOSITION is the offset into the file at
1053                    363  ;  which you wish to position the file pointer.
1053                    364  ; And FROMBYTE is 00 if that offset is from the
1053                    365  ;  file's start, and 01 if that offset is from the
1053                    366  ;  current position.
1053                    367  ;---------------------------------------------------
1053 20 64 0D           368           JSR GETFD2
1056 20 F8 0A           369           JSR PULLADR
1059 8D BF B5           370           STA FMOFFSET
105C 8E C0 B5           371           STX FMOFFSET+$01
105F 20 C4 0B           372           JSR ZEROREC
1062 20 F8 0A           373           JSR PULLADR
1065 F0 14              374           BEQ COMSEEK
1067 A0 1B              375           LDY #$1B
1069 B1 08              376           LDA (FCURWK),Y
106B 18                 377           CLC
106C 6D BF B5           378           ADC FMOFFSET
106F 8D BF B5           379           STA FMOFFSET
1072 C8                 380           INY
1073 B1 08              381           LDA (FCURWK),Y
1075 6D C0 B5           382           ADC FMOFFSET+$01
1078 8D C0 B5           383           STA FMOFFSET+$01
107B                    384  COMSEEK:
107B A9 0A              385           LDA #$0A
107D 8D BB B5           386           STA FMOPCODE
1080 4C D5 0D           387           JMP FMCALLX
1083                    388  ;
1083                    392  ;---------------------------------------------------
1083                    393  BFREWIND:
1083                    394  ; Calling format:  JSR CALLFMLI
1083                    395  ;                  BYT FREWIND
1083                    396  ;                  ADR FILEDESCRIPTOR
1083                    397  ; Sets the file pointer to the beginning of the file.
1083                    398  ; Same as FSEEK(fd,0,0).
1083                    399  ;---------------------------------------------------
1083 20 75 0D           400           JSR GETFDO
1086 20 C4 0B           401           JSR ZEROREC
1089 20 CD 0B           402           JSR ZEROOFF
108C 4C 7B 10           403           JMP COMSEEK
108F                    404  ;
108F                      9  ;---------------------------------------------------
108F                     10  BFCLOSE2:
108F                     11  ; Calling format: JSR CALLFMLI
108F                     12  ;                 BYT FCLOSE2
108F                     13  ; Closes all open files.
108F                     14  ;---------------------------------------------------
108F AD 05 0A            15           LDA HEAPADR          ;locate highest assigned
1092 85 FC               16           STA VP               ; file descriptor.
```

```
1094 AD 06 0A     17            LDA HEAPADR+$01
1097 85 FD        18            STA VP+$01
1099 A0 00        19            LDY #$00
109B B1 FC        20            LDA (VP),Y
109D A8           21            TAY
109E B1 FC        22  ^0        LDA (VP),Y          ;is it allocated, in-use?
10A0 F0 11        23            BEQ >2              ; no skip to next.
10A2 98           24            TYA
10A3 48           25            PHA
10A4 0A           26            ASL                 ;denormalize.
10A5 0A           27            ASL
10A6 20 80 0D     28            JSR FDTO2           ;initialize filemanager.
10A9 90 03        29            BCC >1
10AB 4C A9 0B     30            JMP AERRO
10AE 20 C4 10     31  ^1        JSR COMCLOSE        ;close & deallocate.
10B1 68           32            PLA
10B2 A8           33            TAY
10B3 88           34  ^2        DEY                 ;locate next and test
10B4 88           35            DEY                 ; again.
10B5 88           36            DEY
10B6 10 E6        37            BPL <0
10B8 4C DD 0D     38            JMP COMCLCEX
10BB              39  ;
10BB              40  ;-----------------------------------------------------
10BB              41  BFCLOSE:
10BB              42  ; Calling format: JSR CALLFMLI
10BB              43  ;                  BYT FCLOSE
10BB              44  ;                  ADR FILEDESCRIPTOR
10BB              45  ; Closes a file.
10BB              46  ;-----------------------------------------------------
10BB 20 75 0D     47            JSR GETFDO
10BE 20 C4 10     48            JSR COMCLOSE
10C1 4C DD 0D     49            JMP COMCLCEX
10C4              50  ;
10C4              51  COMCLOSE:
10C4 48           52            PHA                 ;save the file descript.
10C5 A9 02        53            LDA #$02            ; $02 = close.
10C7 8D BB B5     54            STA FMOPCODE
10CA 20 FD AA     55            JSR CALLFM          ;call file manager.
10CD 90 05        56            BCC >3
10CF A9 02        57            LDA #ECLOSE         ;Any error is a "can't
10D1 4C A9 0B     58            JMP AERRO           ; close" error.
10D4 68           59  ^3        PLA                 ;Restore file descript.
10D5 A8           60            TAY
10D6 A9 00        61            LDA #$00            ;De-allocate the buffer.
10D8 91 FA        62            STA (AP),Y
10DA 60           63            RTS
10DB              64  ;
10DB              68  ;-----------------------------------------------------
10DB              69  BFGETNAM:
10DB              70  ; Calling format: JSR CALLFMLI
10DB              71  ;                  BYT FGETNAM
10DB              72  ;                  ADR FILEDESCRIPTOR
10DB              73  ; Returns the address of the filename in a,x lo,hi.
10DB              74  ; The name occupies 30 (decimal) bytes, end padded
10DB              75  ;  with spaces.
10DB              76  ;-----------------------------------------------------
10DB 20 75 0D     77            JSR GETFDO
10DE A5 04        78            LDA FCURNP
10E0 A6 05        79            LDX FCURNP+$01
10E2 85 F5        80            STA ASAVE
10E4 86 F6        81            STX XSAVE
10E6 4C DD 0D     82            JMP COMCLCEX
10E9              83  ;
10E9              84  ;-----------------------------------------------------
10E9              85  BFQUIT:
10E9              86  ; Calling format: JSR CALLFMLI
10E9              87  ;                  BYT FQUIT
10E9              88  ; Reconnects the DOS hooks, and rebuilds the DOS
10E9              89  ; buffers. CAUTION: Use only if DOS has NOT been
10E9              90  ; overwritten by either FMLI, the FMLI buffers, or
10E9              91  ; your routines.
10E9              92  ;-----------------------------------------------------
```

```
10E9 20 51 A8     93            JSR $A851
10EC 20 D4 A7     94            JSR $A7D4
10EF 4C DD 0D     95            JMP COMCLCEX
10F2              96  ;
10F2             100  ;-------------------------------------------------
10F2             101  BFCHMOD:
10F2             102  ; Calling format: JSR CALLFMLI
10F2             103  ;                 BYT FCHMOD
10F2             104  ;                 ADR FILENAME
10F2             105  ;                 ADR MODEBYTE
10F2             106  ; Where MODEBYTE is 00 for unlock, 01 for lock.
10F2             107  ; Allocates a buffer. If you wish to lock/unlock a
10F2             108  ; file your presently have open, close it first, or
10F2             109  ; a new buffer will be created, thus decreasing your
10F2             110  ; available free memory.
10F2             111  ;-------------------------------------------------
10F2 20 E0 0B    112           JSR FALLOC
10F5 90 03       113           BCC >0
10F7 4C 9D 0B    114           JMP AERR2
10FA 20 D6 11    115  ^0       JSR COMNAME
10FD 20 F8 0A    116           JSR PULLADR
1100 F0 04       117           BEQ >1
1102 A9 07       118           LDA #$07
1104 D0 02       119           BNE >2
1106 A9 08       120  ^1       LDA #$08
1108 8D BB B5    121  ^2       STA FMOPCODE
110B 4C D5 0D    122           JMP FMCALLX
110E             123  ;
110E             127  ;-------------------------------------------------
110E             128  BFUNLINK:
110E             129  ; Calling format: JSR CALLFMLI
110E             130  ;                 BYT FUNLINK
110E             131  ;                 ADR FILENAME
110E             132  ; Unlinks (deletes) the named file. NOTE, same warning
110E             133  ; reguarding buffers as FCHMODE.
110E             134  ;-------------------------------------------------
110E 20 E0 0B    135           JSR FALLOC
1111 90 03       136           BCC >0
1113 4C A3 0B    137           JMP AERR1
1116 20 D6 11    138  ^0       JSR COMNAME
1119 A9 05       139           LDA #$05
111B 8D BB B5    140           STA FMOPCODE
111E 4C D5 0D    141           JMP FMCALLX
1121             142  ;
1121             143  ;-------------------------------------------------
1121             144  BFRENAME:
1121             145  ; Calling format: JSR CALLFMLI
1121             146  ;                 BYT FRENAME
1121             147  ;                 ADR OLDNAME
1121             148  ;                 ADR NEWNAME
1121             149  ; Where OLDNAME is a standard filename with optional
1121             150  ; slot, drive, and volume specifications.
1121             151  ; NEWNAME must be a NAME ONLY, padded with trailing
1121             152  ; spaces, and occupying 30 (decimal) bytes.
1121             153  ; NOTE: same warning concerning buffers as FCHMODE.
1121             154  ;-------------------------------------------------
1121 20 E0 0B    155           JSR FALLOC
1124 90 03       156           BCC >0
1126 4C 9D 0B    157           JMP AERR2
1129 20 D6 11    158  ^0       JSR COMNAME
112C 20 F8 0A    159           JSR PULLADR
112F A5 FA       160           LDA AP
1131 8D BD B5    161           STA FMRECORD
1134 A5 FB       162           LDA AP+$01
1136 8D BE B5    163           STA FMRECORD+$01
1139 A9 09       164           LDA #$09
113B 8D BB B5    165           STA FMOPCODE
113E 4C D5 0D    166           JMP FMCALLX
1141             167  ;
```

```
1141                    171  ;-------------------------------------------------------
1141                    172  BFCATL:
1141                    173  ; Calling format: JSR CALLFMLI
1141                    174  ;                 BYT FCATL
1141                    175  ; Standard long catalog listing.
1141                    176  ; Uses last accessed slot, drive, and volume unless
1141                    177  ; preceeded with a call to FSETSDV. NOTE: Same warn-
1141                    178  ; ing about buffers as BFCHMODE.
1141                    179  ;-------------------------------------------------------
1141 20 E0 0B           180          JSR FALLOC
1144 90 03              181          BCC >0
1146 4C A9 0B           182          JMP AERR0
1149 20 94 0C           183  ^0      JSR SWAPUP
114C A9 06              184          LDA #$06
114E 8D BB B5           185          STA FMOPCODE
1151 4C D5 0D           186          JMP FMCALLX
1154                    187  ;
1154                    191  ;-------------------------------------------------------
1154                    192  BFCAT:
1154                    193  ; Calling format: JSR CALLFMLI
1154                    194  ;                 BYT FCAT
1154                    195  ; Produces a listing of file names only. Listed in 2
1154                    196  ; collums with 40 col screen, and 4 collums with 80
1154                    197  ; col. screen. Only the first 19 characters of the
1154                    198  ; name are printed. NOTE: Same warning about buffers
1154                    199  ; as with BFCHMODE.
1154                    200  ;-------------------------------------------------------
1154 20 E0 0B           201          JSR FALLOC
1157 90 03              202          BCC >0
1159 4C A9 0B           203          JMP AERR0
115C 20 94 0C           204  ^0      JSR SWAPUP
115F A9 00              205          LDA #$00
1161 85 0E              206          STA R0
1163 A9 0B              207          LDA /MAPFMERR-$01
1165 48                 208          PHA
1166 A9 B0              209          LDA #MAPFMERR-$01
1168 48                 210          PHA
1169 BA                 211          TSX
116A 8E 9B B3           212          STX $B39B
116D 20 6A AE           213          JSR $AE6A
1170 20 DC AB           214          JSR $ABDC
1173 20 F7 AF           215          JSR $AFF7
1176 20 D1 11           216          JSR CROUT
1179 18                 217          CLC
117A 20 11 B0           218  BCNXTSEC JSR $B011
117D B0 38              219          BCS BCDONE
117F A2 00              220          LDX #$00
1181 8E 9C B3           221  BCLOOP  STX $B39C
1184 BD C6 B4           222          LDA $B4C6,X
1187 F0 2E              223          BEQ BCDONE
1189 30 25              224          BMI >3
118B E8                 225          INX
118C E8                 226          INX
118D E8                 227          INX
118E A0 12              228          LDY #!18
1190 BD C6 B4           229  ^1      LDA $B4C6,X
1193 20 ED FD           230          JSR $FDED
1196 E8                 231          INX
1197 88                 232          DEY
1198 10 F6              233          BPL <1
119A E6 0E              234          INC R0
119C A5 0E              235          LDA R0
119E C9 04              236          CMP #$04
11A0 D0 09              237          BNE >2
11A2 20 D1 11           238          JSR CROUT
11A5 A9 00              239          LDA #$00
11A7 85 0E              240          STA R0
11A9 F0 05              241          BEQ >3
11AB A9 A0              242  ^2      LDA #" "
```

```
11AD 20 ED FD   243           JSR $FDED
11B0 20 30 B2   244  ^3       JSR $B230
11B3 90 CC      245           BCC BCLOOP
11B5 B0 C3      246           BCS BCNXTSEC
11B7 20 D1 11   247  BCDONE    JSR CROUT
11BA 4C DD 0D   248           JMP COMCLCEX
11BD            249  ;
11BD            253  ;-------------------------------------------------------
11BD            254  BFEOF:
11BD            255  ; Calling format: JSR CALLFMLI
11BD            256  ;                  BYT FEOR
11BD            257  ; Tests for end of file. Returns carry clear if either
11BD            258  ; error was caused by end-of-data or a-reg=00.
11BD            259  ; Otherwise returns carry set.
11BD            260  ;-------------------------------------------------------
11BD A5 F5      261           LDA ASAVE
11BF F0 0D      262           BEQ >1
11C1 AD C5 B5   263           LDA FMRETURN
11C4 C9 05      264           CMP #$05
11C6 F0 06      265           BEQ >1
11C8 20 E8 0A   266           JSR BFSEC
11CB 4C CB 0A   267           JMP BFEXIT
11CE 4C DD 0D   268  ^1       JMP COMCLCEX
11D1            269  ;
11D1            273  ;-------------------------------------------------------
11D1            274  CROUT:
11D1            275  ; Send a carriage return to the standard output.
11D1            276  ;-------------------------------------------------------
11D1 A9 8D      277           LDA #$8D
11D3 4C ED FD   278           JMP $FDED
11D6            279  ;
11D6            280  ;-------------------------------------------------------
11D6            281  COMNAME:
11D6            282  ; A general subroutine to fetch the filename and
11D6            283  ; condition the file manager paramater list for any
11D6            284  ; routines requiring an OPEN type format.
11D6            285  ;-------------------------------------------------------
11D6 20 F8 0A   286           JSR PULLADR
11D9 20 B6 0C   287           JSR PARSENAM
11DC 20 94 0C   288           JSR SWAPUP
11DF 20 C4 0B   289           JSR ZEROREC
11E2 60         290           RTS
11E3            291  ;
11E3            292           END
11E3            293  ; end of source listing for FMLI
```

The Call...

BY KATHY RUDOVSKY

I remember when the CALL was a newspaper. In Applesoft it causes the execution of a machine language subroutine at the decimal address where the program begins. Phew!!!! What does that mean?#!!?

Have you ever typed CATALOG <Ret> and sat there looking at the flashing cursor - try it again - and nothing. Somehow, quite mysteriously DOS has been disconnected. Instead of shutting down the system or entering PR#6, try instead using CALL -25153. Keep this one handy, it's a gem.

Marketing Notes

-Don Scellato

I would like to thank everyone who volunteered to help at the Club Booth at the West Coast Computer Faire. The list of volunteers is really long, but each one should still be thanked.

They were Joe Bianco, Chris Chung, Ed Suttles, Ben Lai, Dixon Lee, Kathy Rudovsky, Ruth Wells, Larry Nebron, Dudley Ackerman, Mike Iverson, Allan Woods, Hannelore Maddox, Bill Stokes, Neal Mindell, Joyce Conklin, Carl, Butz, Carol Stender, Steve Booker, Angela Allberry, Stan Guidero, Dave Dresser, Nick Topolos, Jerry Matsukado, Susan Matsukado, Stan Dinsmore, Jeff Menscher, and Phil Bernheim.

We sold enough product and memberships to pay for the booth and the materials sold. Thanks to everyone who helped us do it.

The April Disk of the Month will feature a number of educational programs. Look for it at the meeting.

The May Disk of the Month will feature a number of printer control programs for the Epson and NEC printers. If you have any routines that you've written for the Imagewriter or any other printer, send them to me on disk care of the clubhouse and I will see that you get a free blank disk or a disk of SFAC programs.

Joe Bianco has informed me that his business is doing so well that he wont be able to devote the amount of time he has given during the past two years to running the Sales Table.

Our marketing efforts would not have been as successful as they were during that period of time if we didn't have Joe's help. On behalf of the entire membership and the Board of Directors, I would like to thank Joe for all of the time and help that he's given to us.

I hope to have something more to say in this regard at the May meeting. In the meantime, if any of the readers of this article have eight to ten hours a month available to help, we could use a full time Sales Table manager.

The job entails calling the people who work at the table, getting the goods from the clubhouse to the meeting, setting up the table, selling product, answering questions, closing the table, and returning the product to the clubhouse.

Steve Booker, Larry Nebron, and Chris Chung have been very helpful in regard to all of these tasks and will continue to help in the future. Until a new manager is found, they will trade off as table manager.

We will participate in the Northwest Computer Swap Meet in Oakland in May. Sign up sheets will be posted at the April meeting.

We plan to run a large Swap Meet of our own in September on the date of our September general meeting. This is a good opportunity to sell any used products you no longer require and to purchase some bargains.

If you know of any dealers who would like to participate in this event, please give me their names and I will contact them.

If you are interested in a
C language
interest group, contact:

Bryan Costales

641-7929

```
##########################################
#                                        #
#          AUTHORS WANTED                #
#                                        #
#  Share your experiences, Apple         #
#  programming hints,pascal tips,        #
#  software reviews. Send your           #
#  material to the Cider Press           #
#  Editor at the Clubhouse.....          #
#                                        #
##########################################
```

MACHINATION

by DON SCELLATO

The header you see over this column introduces a new column called MACHINATIONS. This column will be devoted to material about the Macintosh, Macintosh software, and Macintosh compatible hardware.

It will also deal with the activities of the San Francisco Apple Core's Macintosh Interest Group and Lisa Interest Group. The Board of Directors of the San Francisco Apple Core has decided to use all of the resources of the club to support interest groups for each of these 32 computers.

Interest Group leaders have been found to conduct meetings. Macintosh Interest Group meetings will be held at Fort Mason, Building C, immediately after the completion of the April and May General Meetings of the club. Lisa meetings haven't been scheduled yet, but will be announced at the April and May meetings.

Alan Smerican and I will jointly lead the Macintosh Interest Group. If the turnout is as strong as we suspect it will be, meetings may be started in the East Bay as well as in San Francisco. Larry Clopp will lead the Lisa Interest Group. He is receiving a list of about 15 or 20 names of core members of this group and will begin to hold meetings at a place and date of his choice.

His phone number will be published in the next issue of the Cider Press. In the meantime, if any of the readers must contact him, leave a message at the clubhouse and it will be passed on.

The first meeting of the Macintosh Interest Group will be held at 1:00 P.M. at Building C, Room 100, Fort Mason on April 14. A member of the Apple Computer Macintosh Team will present and discuss MacDraw, MacTerminal, and one other piece of software for the Macintosh.

The next meeting of this interest group will be held on Saturday, May 19, at Fort Mason, Building C at 1 P.M. The topic of discussion for this meeting will be announced at the first meeting and will be placed on the answering machine tape at the clubhouse.

If you have any questions about these meetings or any ideas for discussion, call Alan Smerican or me at the phone numbers listed in the Cider Press.

In the future this column will be devoted to articles about the Macintosh. If any reader has anything to share or wishes to write an article for the column, I would be happy to relinquish the byline for that of a guest author anytime.

If you've got something to share, please send it to me at the clubhouse. Anyone who contributes an article which is published in the Cider Press will receive either a free 5 1/4" disk in return for the disk on which the article is sent (written with Applewriter, MacWrite, Wordstar, Apple Pie, or Word Handler) or will receive a free 3 1/2" disk.

Just make your choice when you submit the article. The same offer applies to general articles about the Macintosh which are intended for use in the Cider Press instead of use in this column. Barring any inspiring contributions from other sources, the next edition of this column will deal with Macintosh file handling procedures.

VisiCalc Interest Groups

-Don Scellato

The San Francisco Spreadsheet Interest Group met on Tuesday, March 13, at the clubhouse at 7:30 P.M. We spent time dealing with a few troublesome boolean logic statements.

The next meeting of the San Francisco group will be held at the clubhouse on Tuesday, April 10. Phil Berheim will demonstrate SpreadLink, a Dow Jones Stock Management/Communications program.

This meeting will be held at 7:30 P.M. The May meeting of this group will be held on Tuesday, May 8 at 7:30 P.M. at the clubhouse. We will spend some time on the DOS version of Multiplan.

The East Bay Spreadsheet Interest Group met at Breuner's Furniture Store in Pleasant Hill at 7:00 P.M. on Monday, March 26. The meeting was spent working with the DOS 3.3 version of Multiplan, converting VisiCalc templates to Multiplan templates, and using VisiCalc to control the type size and line spacing of an Epson MX-100 printer.

The next meeting of this group will be held on Thursday, April 26 at 7:00 P.M. at Breuner's off of Oak Park Road. The meetings are held in the second floor conference room of the store. This meeting will deal with checkbook reconciliation and with the Macintosh version of Multiplan.

Programming Contest

by
Don Scellato

Since the Programming Contest of 1983 was successful as regards having members contribute programs and having members win really great prizes, we've decided to try again.

I've already talked to several magazines, software publishers, and hardware manufacturers about contributing prizes for the contest. A number of them have already given the club something to use as a prize or have promised a prize to the club.

For example, Videx, Inc., has donated an Ultraterm Video Expansion Board to the club to use as a prize. The Apple Orchard has once again indicated that they would give each contest entrant a free one year subscription to the magazine.

Mark Pielzarski of Penguin Software has indicated that he would consider donating a prize. I will follow up with him. Roger Wagner of Roger Wagner Publishing has also indicated that he will consider donating a prize or two. Watch the Cider Press for more informatio on the prizes.

The Programming Contest will be structured much as it was last year wioth prizes being awarded for Programming Utilities, Educational Software, Home Applications, Management Programs, Games and Entertainment, Graphics Utilities, and a special catgegory for Macintosh Applications.

The Contest will open on July 1, 1984, and it will close on October 31, 1984. (Happy Halloween....) All entries must be submitted by a club member.

Entries must be accompanied by a Contest Entry Form which is currently available from the clubhouse by sending a stamped, self-addressed envelope to the clubhouse or by phoning during office hours and asking for an entry form. Forms will also be available at each meeting starting with the May General Meeting.

If you've got some favorite application you've written and would like to share it with other members of the Club and maybe win a super prize, enter the contest. Watch for more information in the next issue of the Cider Press.

NUERS

-Kathy Rudovsky

The March session of Nuers welcomed guest speaker Leroy Larson. Leroy was among the first 35 original Apple Core members and shared with the group some of the club's foundations. The main topic was the S.F.A.C. Library which Leroy contributed to immensely in its assemblance.

In the beginning.....(6 or so years ago) few commercial programs were in existence, so the members would gather monthly to share and exchange programs they had written or obtained from friends or other User Groups. The Library today contains a vast number of programs, all of which are public domain.

Leroy highly recommends that in using the library programs that you list and modify them to suit your own needs for these programs are not only useful, but excellent learning tools. They will help you to expand your knowledge of your computer system. (The old trial and error routine), although a time consuming and sometimes frustrating approach, it truly does stimulate the imagination and provide an everlasting educational experience.

For those times when frustration does get the better of you, The Bit Banger, a foam rubber mallet conceived in San Francisco by Deborah Grosshandler, might prove useful in alleviating "Nuer's Stress".

Most everyone present took home the August 1981 Disk of the Month which contains the data base program "File Cabinet". This program was first distributed by Apple Computer with the purchase of an Apple.

Since then it has been modified several times. It has many features that are quite useful and is fairly easy to set up. At least it doesn't come with a 500 hundred page manual.

The best way to review the Library is to scan through the documentation booklets. Don Scellato announced that the 1982-83 Program Library should become available to members by May, however, the documentation booklet will not be available for another month or so.

DID YOU KNOW? <ESC> F clears the screen from the point of the cursor to the bottom of the screen. ?Half way HOME?

Poke 33,33 Scrunches your program list for easier editing. It also directs your printer to type out your listing 80 characters across the page.
Poke 33,40 Brings your screen back to normal - easier yet - TEXT does the same thing.

Variables are a programmers term which designate a memory location used to hold a value. String variables hold a literal value between quotation marks. For more information on variables and how to use them in a program refer to the Green Apples group instructed by Keith Thomas.

NUER'S rendezvous on the third Wednesday of each month at 7:30 pm at the SFAC Clubhouse, 1515 Sloat Blvd. #2. Anyone interested or curious is wecome to attend. The atmosphere is informal and friendly.

INTRO TO THE APPLE COMPUTER

A group of introductory mini-seminars are scheduled for May and June. These seminars are designed to provide exposure to the varied capabilities of the Apple Computer.

Sign-up sheets will be posted at the April & May General Meetings or contact the S.F.A.C. Clubhouse 566-2342 or Kathy Rudovsky 756-6480.

All Seminars are scheduled for 7:30 pm at the S.F.A.C. Clubhouse 1515 Sloat Blvd. Suite 2.

Monday	May	21	- Logo
Wednesday	May	23	- Sound & Music
Tuesday	May	29	- Assembly Language
Wednesday	May	30	- Applesoft Basic
Monday	June	4	- Bus. Applications
Wednesday	June	6	- Forth
Monday	June	11	- Games
Thursday	June	14	- Ed. Applications
Monday	June	18	- Pascal
Wednesday	June	20	- Word Processing
Thursday	June	21	- Graphics
Monday	June	25	- Telecommunications

Insufficient participation will constitute cancellation of individual seminars. Contact the S.F.A.C. Clubhouse or Kathy Rudovsky three days prior for confirmations.

May

Monday	Tuesday	Wednesday	Thursday
	1 Forth* 7:30 East Bay** 7:30		3 Green Apples* 8:00
	8 VisiCalc* 7:30 Marin CP/M**	9 Board Mtg.*	
14 Pascal** 7:30	15 Forth* 7:30	16 Nu'ers* 7:30	
21 Intro-LOGO*	22 Dot Matrix* Printers Marin CP/M**	23 Sound & Music*	24 Green Apples*
	29 Assembly Lng*	30 S.F.CP/M*	31 Green Apples*

June

Monday	Tuesday	Wednesday	Thursday
4 App.Basic* 7:30	5 Forth* 7:30 East Bay** 7:30	6 Intro-Forth* 7:30	
11 Games*	12 Spreadsheets* 7:30 Marin CP/M**	13 Board Mtg.*	14 Ed. App.*
18 Intro-Pascal* 7:30	19 Forth* 7:30	20 Nu'ers* 7:30	21 Intro-Graphics*
25 Pascal** 7:30 Intro-Telecomm.**	26 Dot Matrix* Printers Marin CP/M**	27 S.F. CP/M*	28 Intro - Bus. App.*

* At the ClubHouse.

** ElseWhere.

INTEREST GROUP LEADERS

Marin CP/M	Jim Ayers	383-0473(day) 383-5759(eves)
S.F. CP/M	Ron Dell'Aquila	566-8460
East Bay	Molly Mitchell	526-9541
Green Apples	Keith Thomas	837-3164
Dot-Matrix Printers	Michael Fischer	557-1250(day) 933-4435(eves)
Forth	Dudley Ackerman	626-6295
Marin Graphics	Reed Harris	924-7221
	Bob Sachs	332-3331
Nu'ers	Kathy Rudovsky	756-6480
Pascal	Mike Schnapp	386-2238
VisiCalc et al	Don Scellato	672-2051
Lisa	Larry Clopp	333-3707(eves)
Macintosh	Alan Smerican/Don Scellato	566-2342
S.F.A.C. Clubhouse	* * *	566-2342

SF APPLE CORE
1515 Sloat Blvd., Suite 2
San Francisco, CA. 94132

TIME VALUE

Bulk Rate
U.S. Postage
PAID
Permit #85
El Cerrito, CA 94530

TO:

3121AUG84
DOUGLAS STORKOVICH
921 PINTO CT
WALNUT CREEK CA 94596